M14 Apply: Textbook/Programming Assignment(s) –

Final Project, Part 4, Validation Testing

**Data Set One:** Negative Numbers in the Quantity Fields (*txtBeefDogs.Text*, *txtPorkDogs.Text*, and *txtTurkeyDogs.Text*)

Negative numbers do not throw an error but will skew the hotdog quantity in any future database connection. Negative numbers will also skew Subtotal, and all subsequent calculations for charging the customer.

To prevent this behavior, provide an IF statement that will do the following:

* Throw a MessageBox error
* Reset the textbox entries back to a default of 0
* Exit the btnCalculate\_Click Sub so the CPU won’t do useless calculations

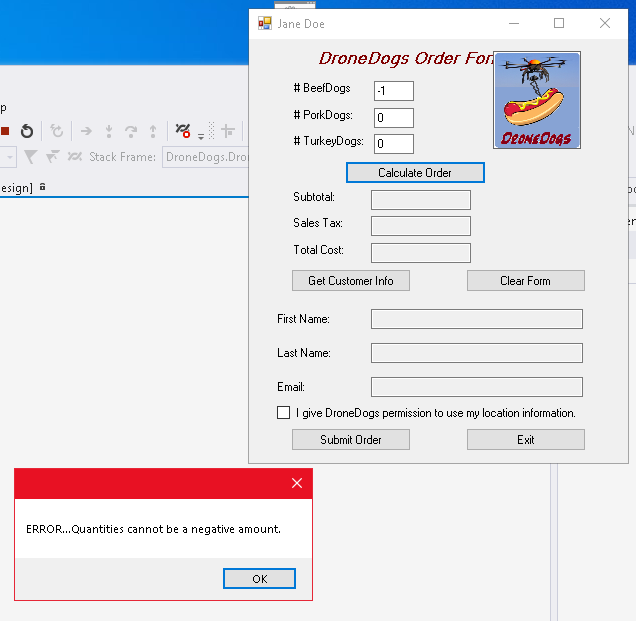


Figure 1-7, If Statement Error Handling in action

**--------------------------------------------------------------------------------------------------------------------------------------------------**

**Data Set Two:** Null & non-numeric data in the Quantity Fields (*txtBeefDogs.Text*, *txtPorkDogs.Text*, and *txtTurkeyDogs.Text*)

Null “values” and non-numeric data will throw an exception called *System.FormatException* upon clicking Calculate Order which calls Sub *btnCalculate\_Click* .

To avoid *System.FormatException*, do the following:

* Try,Catch,Finally Error Handling for each Quantity Fields (*txtBeefDogs.Text*, *txtPorkDogs.Text*, and *txtTurkeyDogs.Text*) within Sub *btnCalculate\_Click* .

To ease user burden, do the following:

* Pre-populate the aforementioned fields with “0” inside the Sub DroneDogsOrder\_Load
* Insert 0 in each field anytime a Catch is triggered, whether it be Null or non-numeric based.

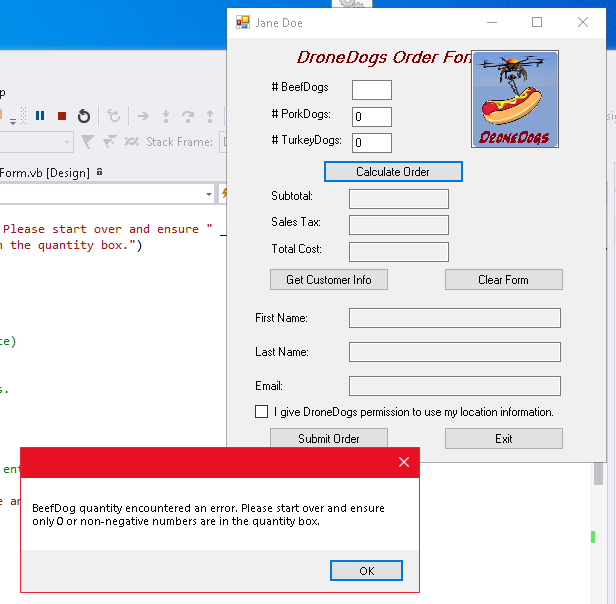


Figure 2-7, If Statement Error Handling in action

**--------------------------------------------------------------------------------------------------------------------------------------------------**

**Data Set Three:** Non-numeric data is automatically inserted into Quantity Fields (*txtBeefDogs.Text*, *txtPorkDogs.Text*, and *txtTurkeyDogs.Text*) whenever the Clear Form button is pressed which calls Sub *btnClear\_Click* .

Null & non-numeric data in the Quantity Fields (*txtBeefDogs.Text*, *txtPorkDogs.Text*, and *txtTurkeyDogs.Text*)

Null “values” and non-numeric data will throw an exception called *System.FormatException* upon clicking Calculate Order which will call the Sub *btnCalculate\_Click* .

To remove the non-numeric data, do the following:

* Try,Catch,Finally Error Handling for each Quantity Fields (*txtBeefDogs.Text*, *txtPorkDogs.Text*, and *txtTurkeyDogs.Text*).
* Pre-populate the aforementioned fields with “0” with Sub *btnClear\_Click* to be “0”.

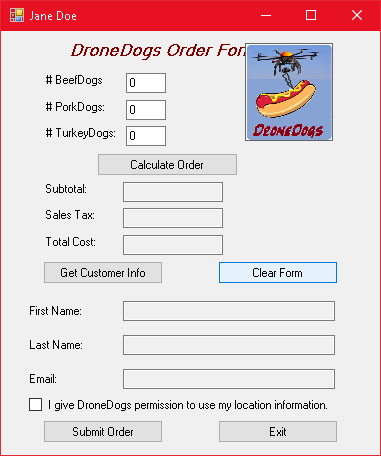


Figure 3-7, Pre-populated, and corrected .Text fields after clicking “Clear Form”

**--------------------------------------------------------------------------------------------------------------------------------------------------**

**Data Set Four:** Messagebox on Exit button displays the wrong company name.

Clicking the Exit button calls Sub *btnExit\_Click* which returns a *MessageBox ("Thank you for ordering from the International House of Cards")* .

To correct this, do the following:

* *MessageBox ("Thank you for ordering from DroneDogs!")* .
* Alternatively, just remove the MessageBox altogether, because exiting the program does not imply purchasing.

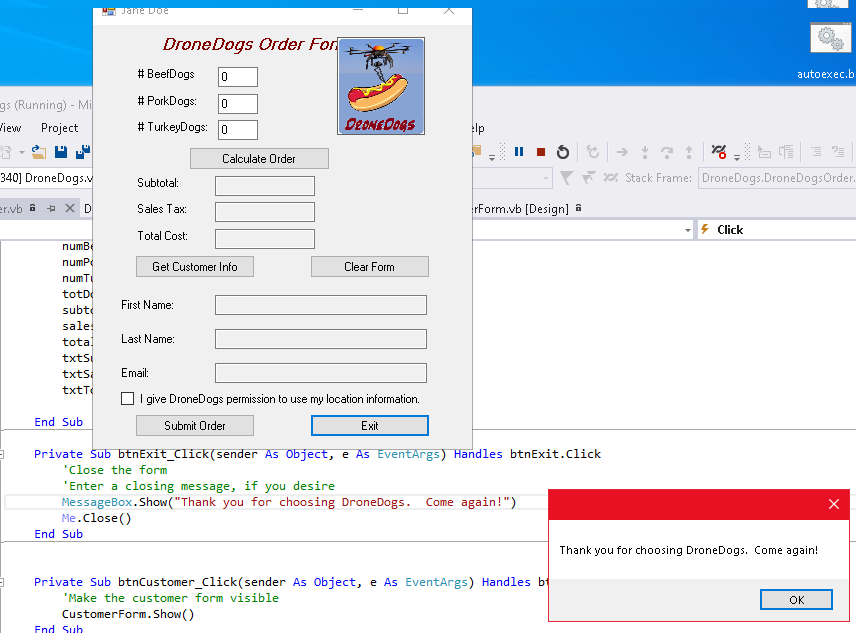


Figure 4-7, Corrected Exit message after clicking “Exit”

**--------------------------------------------------------------------------------------------------------------------------------------------------**

**Data Set Five:** Logo covers the order form text

The application window is too small, and the logo is covering the text “DroneDogs Order Form”

To correct this, do the following:

* In Design mode, enlarge the horizontal application window.
* Drag the DroneDogs logo to the right, thus unobstructing the order form text.



Figure 5-7, Corrected application size and logo placement

**--------------------------------------------------------------------------------------------------------------------------------------------------**

**--------------------------------------------------------------------------------------------------------------------------------------------------**

**Data Set Six:** Subtotals should not be <= 0 will allow customer to process orders that have 0 quantities

The subtotal should not be <= 0. This implies that nothing is being ordered (which defeats the purpose of clicking the button).

To correct this, do the following:

* Provide an IF statement after subtotal = totDogs \* COST\_PER\_DOG , that will that will do the following btnCalculate\_Click
* Throw a MessageBox error
* Exit the btnCalculate\_Click Sub so the CPU won’t do useless calculations

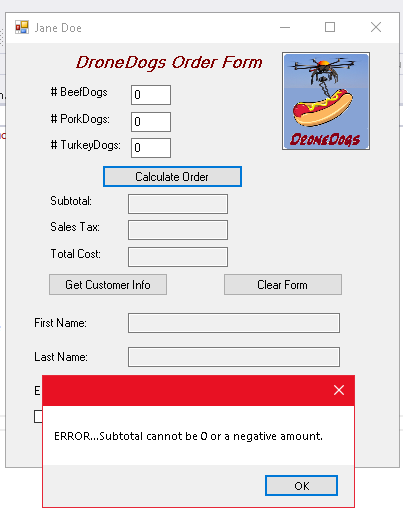
****

Figure 6-7, Subtotal error message

**--------------------------------------------------------------------------------------------------------------------------------------------------**

**Data Set Seven:** Previous order does not clear after submitting the order.

The previous order should reset back to default so that placing a new order is easier, and resetting the location permission checkbox.

To correct this, do the following:

* Call the Sub *btnClear\_Click* from the end of Sub *btnSubmit\_Click.*
* Uncheck the checkbox

Figure 7-7, Subtotal error message

**--------------------------------------------------------------------------------------------------------------------------------------------------**

**Data Set Eight:** # BeefDogs label inconsistent with other order labels.

lblBeefDogs text is missing a colon at the end which makes it inconsistent with the other order labels.

To correct this, do the following:

* Add a colon to the text properties of *lblBeefDogs.*
* Uncheck the checkbox

**--------------------------------------------------------------------------------------------------------------------------------------------------**